

00862.023348.

PATENT APPLICATION

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of:)
MIYUKI ENOKIDA ET AL) Examiner: Not Yet Assigned
Application No.: 10/729,007) Group Art Unit: Not Yet Assigned
Filed: December 8, 2003)
For: ENCODED DATA GENERATION)
METHOD AND APPARATUS,)
AND IMAGE PROCESSING)
METHOD AND APPARATUS) March 9, 2004

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

SUBMISSION OF PRIORITY DOCUMENTS

Sir:

In support of Applicants' claim for priority under 35 U.S.C. § 119, enclosed are certified copies of the following Japanese applications:

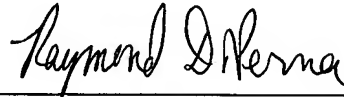
2002-356738, filed December 9, 2002;

2003-176930, filed June 20, 2003; and

2003-201162, filed July 24, 2003.

Applicants' undersigned attorney may be reached in our New York office by telephone at (212) 218-2100. All correspondence should continue to be directed to our address given below.

Respectfully submitted,



Attorney for Applicants

Registration No. 44,063

FITZPATRICK, CELLA, HARPER & SCINTO
30 Rockefeller Plaza
New York, New York 10112-3800
Facsimile: (212) 218-2200

NY_MAIN 412495v1



CFM03348
US

日本国特許庁
JAPAN PATENT OFFICE

Appln. No. 10/729,007
GIAU: NYA

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出願年月日 2002年12月9日
Date of Application:

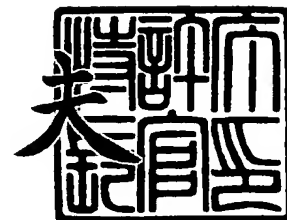
出願番号 特願2002-356738
Application Number:
[ST. 10/C]: [JP 2002-356738]

出願人 キヤノン株式会社
Applicant(s):

2004年 1月 6日

特許庁長官
Commissioner,
Japan Patent Office

今井 康



出証番号 出証特2003-3108698

【書類名】 特許願

【整理番号】 226615

【提出日】 平成14年12月 9日

【あて先】 特許庁長官殿

【国際特許分類】 G06F 3/00

【発明の名称】 符号データ作成方法

【請求項の数】 1

【発明者】

 【住所又は居所】 東京都大田区下丸子 3 丁目 3 0 番 2 号 キヤノン株式会社
社内

 【氏名】 榎田 幸

【特許出願人】

 【識別番号】 000001007

 【氏名又は名称】 キヤノン株式会社

【代理人】

 【識別番号】 100076428

 【弁理士】

 【氏名又は名称】 大塚 康德

 【電話番号】 03-5276-3241

【選任した代理人】

 【識別番号】 100112508

 【弁理士】

 【氏名又は名称】 高柳 司郎

 【電話番号】 03-5276-3241

【選任した代理人】

 【識別番号】 100115071

 【弁理士】

 【氏名又は名称】 大塚 康弘

 【電話番号】 03-5276-3241

【選任した代理人】

【識別番号】 100116894

【弁理士】

【氏名又は名称】 木村 秀二

【電話番号】 03-5276-3241

【手数料の表示】

【予納台帳番号】 003458

【納付金額】 21,000円

【提出物件の目録】

【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【包括委任状番号】 0102485

【ブルーフの要否】 要

【書類名】 明細書

【発明の名称】 符号データ作成方法

【特許請求の範囲】

【請求項 1】 サーバが管理する符号データのうち、断片的な第 1 の符号データを格納した格納手段を備えるクライアントにおいて、J P E G 2 0 0 0 符号データを作成する符号データ作成方法であって、

前記クライアントにおいて前記 J P E G 2 0 0 0 符号データの作成に必要なとなる符号データと、前記格納手段に格納された前記第 1 の符号データとから、不足する第 2 の符号データを算出する算出工程と、

前記サーバに対して、算出された前記第 2 の符号データを要求する要求工程と

、
前記サーバから、前記第 2 の符号データを取得する取得工程と、

取得された前記第 2 の符号データを前記格納手段に格納する格納工程と、

取得された前記第 2 の符号データのヘッダ情報を解析して、前記符号データを複数の独立した符号データに分割する分割工程と、

前記分割工程で分割された単位毎に、前記独立した符号データを構成する全ての符号データが前記格納手段に格納されているか否かを判定する判定工程と、

前記独立した符号データを構成する全ての符号データが格納されていない場合、格納されていない部分にダミー符号データを格納するダミー格納工程と、

前記格納手段に格納された符号データを J P E G 2 0 0 0 符号データとして出力する出力工程と

を有することを特徴とする符号データ作成方法。

【発明の詳細な説明】

【 0 0 0 1 】

【発明の属する技術分野】

本発明は、ネットワークを介して受信した断片的な画像データから符号データを作成する技術に関する。

【 0 0 0 2 】

【従来の技術】

インターネット上では、WWWサーバにWeb（ウェブ）ブラウザからアクセスして文書データや画像データ等の情報を閲覧することが盛んに行われている。このように、インターネット上に情報を公開するWWWサーバと、その情報を閲覧するためのクライアントとを含むシステム環境において、各クライアントではウェブブラウザを使用して、WWWサーバによって公開された情報を閲覧することができる。

【0003】

WWWサーバには、通常、ホームページといわれる公開するための情報をHTMLで記述した文書が保存されており、それにクライアント側のウェブブラウザがアクセスしてクライアント側のコンピュータに表示する。また、クライアント側のウェブブラウザは、表示しているページ内のリンクを辿っていくことにより、必要な情報を得ることができる。さらに、WWWサーバが管理しているファイルをダウンロードする方法として、File Transfer Protocol（以下、「FTP」と略す。）という方法がある。このFTPとは、ネットワークを通して、WWWサーバ上にあるファイルの内容を一度にクライアント側のコンピュータに転送する仕組みのことである。

【0004】

また、サーバが管理する画像ファイルへ断片的にアクセスして、クライアントで表示するためのプロトコルとして、Flashpix/IIPがある。このインターネット・イメージング・プロトコル（IIP）は、Flashpixという画像データファイルフォーマットに最適なプロトコルになっており、ネットワークを通して画像データの一部分を要求できるものである。このときのアクセスは、Flashpixのタイル単位に行われる。

【0005】

一方、このFlashpix/IIPをそのままJPEG2000に適用した場合を考える。ここで、JPEG2000では、各スケーラビリティ（Scalability）の符号データは、そのスケーラビリティより1つ下のスケーラビリティのデータとの差分データで構成されている。そこで、クライアント側で受信した断片的な符号データをキャッシュしておき、全符号データをデコーダに引き渡して最初から

再デコードする方法と、デコーダを途中で止めておいて今回受信した符号データをデコーダに渡し、前回の続きからデコードする方法とがある。

【 0 0 0 6 】

このような方法のうち、マルチ解像度データを有する画像符号データの中から必要な部分のデータのみを取り出して別の符号データに変換する方法が従来から提案されている（例えば、特許文献 1 参照）。

【 0 0 0 7 】

上記特許文献 1 においてソースとなる画像データは、ウェーブレット (Wavelet) 変換、或いはWavelet-likeな変換を用いて、マルチ解像度のデータを管理することが可能な符号データである。そして、このソースとなる符号データの中から、ユーザが選択した空間的な領域のデータを処理するために必要な符号データを取り出し、1つの独立した符号データに変換する方法が記載されている。尚、この時ソースの符号データから取り出される部分的な符号データは、J P E G 2 0 0 0 のコードブロックに対応しており、今回のユーザからの要求を処理するために必要な符号データを含んでいる。

【 0 0 0 8 】

ここで、サーバから送られてくる符号データをクライアント側で最初から再デコードする場合、それらの断片的な符号データをキャッシュのような仕組みを持たないで、直接、J P E G 2 0 0 0 準拠の 1 つの符号データファイルに作り直すことが可能である。

【 0 0 0 9 】

【特許文献 1】

米国特許第 6 0 4 1 1 4 3 号明細書

【 0 0 1 0 】

【発明が解決しようとする課題】

しかしながら、サーバ側から送られてくる断片的な符号データは、クライアント側から要求する順に送られてくるため、クライアント側で受信した符号データを J P E G 2 0 0 0 準拠の 1 つの符号データに変換するためには、クライアント側での煩雑な処理が必要であるという問題点がある。

【 0 0 1 1 】

また、サーバから送られてきた断片的な符号データを独自形式のキャッシュファイルとして構成し、クライアント側の J P E G 2 0 0 0 デコーダが直接このキャッシュファイルをリードすることも可能である。しかし、この場合、J P E G 2 0 0 0 デコーダには独自のキャッシュファイルをリードするための処理が必要となり、処理が複雑になると共に、汎用の J P E G 2 0 0 0 デコーダが使えないという問題点がある。

【 0 0 1 2 】

そこで、J P E G 2 0 0 0 符号データをパケット (packet) 単位で受信し、これらの packet データをキャッシュしておいて、デコーダへ符号データを渡す際に、まだ受信していない packet データ部分に、zero length packet (以下、「Z L P」と略す。) データを挿入し、既に受信した packet データとあわせて、1 つの J P E G 2 0 0 0 準拠の符号ファイルを作成することで、メインヘッダの書き換えなどの煩雑な処理を行わずに、一般的な J P E G 2 0 0 0 デコーダで処理可能な J P E G 2 0 0 0 ファイルを作成することが考えられる。

【 0 0 1 3 】

しかし、受信した全ての packet データを利用して作成された符号データは、クライアント側のアプリケーションを使っているユーザの要求した画像サイズ、画質及び画像領域等において、ユーザの望む表示形態を考慮していない。そのため、このような符号データを受け取ったデコーダでは、その一部のデータから生成されたデコード結果を利用して表示画面が作成されることがあり、キャッシュされている全ての符号データを使って 1 つの J P E G 2 0 0 0 符号データファイルを作ることは、デコーダ自体の処理にも無駄な処理が生じるという問題がある。

【 0 0 1 4 】

さらに、キャッシュされている符号データが多くなるほど、デコーダへ渡す符号データを作成するときに発生するデータのコピー作業が多くなる。これは、クライアントの表示に要するまでの時間が長くなることにつながり、パフォーマンス低下という問題が生じる。特に、画像の拡大やスクロールという命令をユーザが行うことで、表示に直接必要のないキャッシュデータが多くなり、上記の問題

が頻繁に起こることになる。

【0015】

さらにまた、上記特許文献1には、以下のような問題点がある。すなわち、上記特許文献1に記載の技術をネットワークを通した通信であるIIPにそのまま適用した場合、クライアントからの要求毎に全階層の全コードブロックの符号データを送ることになり、クライアント側で既に受信済みの符号データまでも送信することになる。また、送信する符号データのヘッダ部分は、今回要求した画像データの情報（例えば、画像サイズ、階層情報等）に書き換えられてしまい、サーバで管理されている符号データの本来の情報を取得することができない。

【0016】

本発明は、このような事情を考慮してなされたものであり、クライアントにキャッシュされている断片化された符号データと、サーバから必要な分だけ受信した断片化された符号データとを用いて、クライアントにおいて汎用JPEG2000デコーダで使用可能な符号データであって、当該符号データのデコード及び画像データの表示処理を高速に行うことができる符号データを好適に作成することができる符号データ作成方法を提供することを目的とする。

【0017】

【課題を解決するための手段】

上記課題を解決するために、本発明は、サーバが管理する符号データのうち、断片的な第1の符号データを格納した格納手段を備えるクライアントにおいて、JPEG2000符号データを作成する符号データ作成方法であって、

前記クライアントにおいて必要となる符号データと、前記格納手段に格納された前記第1の符号データとから、不足する第2の符号データを算出する算出工程と、

前記サーバに対して、算出された前記第2の符号データを要求する要求工程と、

前記サーバから、前記第2の符号データを取得する取得工程と、

取得された前記第2の符号データを前記格納手段に格納する格納工程と、

取得された前記第2の符号データのヘッダ情報を解析して、前記符号データを

複数の独立した符号データに分割する分割工程と、

前記分割工程で分割された単位毎に、前記独立した符号データを構成する全ての符号データが前記格納手段に格納されているか否かを判定する判定工程と、

前記独立した符号データを構成する全ての符号データが格納されていない場合、格納されていない部分にダミー符号データを格納するダミー格納工程と、

前記格納手段に格納された符号データを J P E G 2 0 0 0 符号データとして出力する出力工程と

を有することを特徴とする。

【0018】

【発明の実施の形態】

以下、図面を参照して、本発明の実施形態について詳細に説明する。

【0019】

<第1の実施形態>

図1は、ネットワーク上に複数のコンピュータが接続されている様子を示す図であり、本発明の第1の実施形態に係るサーバ及びクライアントを含むネットワークシステムの構成を示す概要図である。

【0020】

図1において、100はインターネットに代表されるネットワークである。101、103は、ネットワーク100に接続されているサーバ・コンピュータであり、画像データを送信するためのJ P E G 2 0 0 0用のI I Pサーバを始めとして、WWWサーバ機能に必要なソフトウェアが実行されている。また、大量の画像データも格納されている。また、102a、102bはクライアント・コンピュータであり、ウェブブラウザやJ P E G 2 0 0 0デコーダ等のクライアント側に必要なソフトウェアが実行されている。

【0021】

図2は、図1のサーバ・コンピュータ101、103又はクライアント・コンピュータ102a、102bの各コンピュータシステムのハードウェア構成の一例を示す図である。図2において、201はCPUであり、システム全体の制御等を行っている。202はキーボードであり、マウス201aと共に本システム

に対して情報等を入力するために使用される。また、203は表示部であり、CRTや液晶ディスプレイ等で構成されている。

【0022】

一方、204はROM、205はRAMであり、本システムにおける記憶装置を構成しており、システムが実行するプログラムやシステムが利用するデータ等を記憶する。また、206はハードディスク装置、207はフレキシブルディスクであり、本システムのファイルシステムに使用される外部記憶装置を構成している。さらに、208はプリンタである。さらにまた、209はネットワークを制御する部分であり、ここからネットワーク100（例えば、インターネット等）上に接続されているサーバ・コンピュータ等のネットワーク上の資源にアクセスする。

【0023】

本実施形態では、既に生成済みのJPEG2000ファイルがサーバ・コンピュータで管理されている。一方、クライアント・コンピュータでは、画像表示アプリケーション等を用いてユーザが要求したJPEG2000ファイル内の必要な部分のみを、JPEG2000用IIPを使用して上記サーバ・コンピュータにアクセスし、符号データを断片的に受信し、受信した断片的な符号データをクライアント・コンピュータ上の例えばハードディスク206等にキャッシュする。

【0024】

以下では、ハードディスク206等にキャッシュされた断片的なJPEG2000符号データを、JPEG2000のデコーダに渡すJPEG2000準拠の符号データに変換する部分について説明する。

【0025】

また、ユーザは、Windows（登録商標）マシン等を使用してホームページを開き、そこに書かれているJPEG2000の画像へのリンクをクリックすることで、JPEG2000の画像をユーザの用途に適した画像サイズや解像度で表示するために必要な断片的なデータを取得し、キャッシュする。そして、それらのキャッシュデータから一つのビットストリームを作り出し、デコードし、

表示するという場合を想定する。

【0 0 2 6】

ここで、一般的な J P E G 2 0 0 0 の符号データについて説明する。図 3 は、Layer-resolution level-component-position progression (以下、「S N R Progression」と記す。) に沿って記録された J P E G 2 0 0 0 ファイルの構成を示す概念図である。S N R Progression に準じた場合、layer/resolution/component/position の順に記録される。このような符号データの並び方を規定することは、「progression order」と呼ばれる。

【0 0 2 7】

また、図 4 は、J P E G 2 0 0 0 の解像度スケーラビリティを説明する図である。すなわち、解像度 (画像サイズ) と Resolution 番号との関係を示すための図である。図 4 では、最も小さい解像度の画像の resolution 番号を 0 とし、resolution 番号が 1 つ増加するごとに画像の幅と高さが 2 倍になっていく。また、各 layer 内は、resolution 番号の小さい順にデータが格納されている。そして、Layer 番号は、復元する画像の原画に対する S/N 比に対応しており、layer 番号が小さいほど S/N 比が悪くなる。

【0 0 2 8】

さらに、1 つの J P E G 2 0 0 0 ファイル内での resolution 番号、layer 番号及び component 番号の最大値は、エンコーダによって予め設定されており、そのパラメータに従ってエンコードされており、その情報は符号データの中に格納されている。また各 packet は、その packet 内に格納されている code-block の情報を管理している packet header 部と、各 code-block の符号データとから構成されている。

【0 0 2 9】

このような J P E G 2 0 0 0 符号データを使うことによって、ユーザは、サーバにある全ての画像データを取得する必要がなく、クライアント側で表示等するために必要な部分の符号データのみをサーバから受信すればよい。尚、ユーザ側 (クライアント・コンピュータ) の受信データの単位としては、J P E G 2 0 0 0 の packet、或いは packet よりも更に小さい符号化単位であるコードブロック単

位が考えられる。本実施形態では、ユーザがサーバから受信するデータ単位としてpacket単位を想定する。

【0 0 3 0】

図5は、第1の実施形態におけるサーバとクライアント間でのpacket単位のデータのリクエスト及びレスポンスを説明するための概念図である。

【0 0 3 1】

図5において、クライアント501はサーバ502に画像のタイル番号、resolution levelと、layer、component、position番号を指定して必要なデータを要求する。そして、サーバ502は、画像503のコードストリームを解析して、指定された画像のタイル番号、resolution levelとlayer、component、position番号に相当するpacketデータを抜き出してクライアント501に送り返す。

【0 0 3 2】

次に、図6A及び図6Bを用いて本実施形態で使用するサーバ・コンピュータで管理されているJ P E G 2 0 0 0ファイルの一例について説明する。図6Aは、サーバ・コンピュータに格納されている原画像データをタイルに分割した例を示す図である。図6Aでは、縦横とも各々4つのタイル、すなわち画像全体で16個のタイルに分割した例が示されている。

【0 0 3 3】

また、図6Bは、図6Aで示したタイル分割された画像データに関するJ P E G 2 0 0 0符号データ（すなわち、当該画像データを「S N R Progression」でエンコードしたデータ）のデータ構造を示す図である。図6Bに示される符号データは、layerは0～2の3種類（Lay0、Lay1、Lay2）、resolution levelは、0～3の4種類（Res0、Res1、Res2、Res3）、componentとpositionは各1種類（Com0、Pos0）である場合のJ P E G 2 0 0 0で符号化した時の符号データの構成例である。

【0 0 3 4】

例えば、J P E G 2 0 0 0の最大解像度の画像サイズを1 0 2 4 × 1 0 2 4画素とした場合、各タイルの画像サイズは2 5 6 × 2 5 6画素となる。また、各タイルが4種類のresolution levelを持つ場合の各解像度での画像サイズは、2 5

6 × 2 5 6、1 2 8 × 1 2 8、6 4 × 6 4、3 2 × 3 2 画素となる。

【0 0 3 5】

さらに、J P E G 2 0 0 0 符号データ内のメインヘッダ部分については、図 6 B に示すように、画像サイズを示すパラメータである Xsiz、Ysiz が共に 1 0 2 4 であり、画像全体の画像サイズを示している。一方、タイルのサイズを示す X T siz、Y T siz は、上述したタイルのサイズである 2 5 6 の値を持っている。これらのパラメータにより、この J P E G 2 0 0 0 符号データは、1 6 個のタイルに分割された符号データを有することになる。そして、これらの各タイルの符号データが、図 6 B に示すように「Tile part Header」で区切られて格納されている。

【0 0 3 6】

図 7 は、第 1 の実施形態におけるクライアント・コンピュータ上のアプリケーションでの処理手順の概要を説明するためのフローチャートである。本実施形態においては、クライアント側では、J P E G 2 0 0 0 画像表示アプリケーションが起動しており、図 7 に示されるフローチャートは、表示したい J P E G 2 0 0 0 ファイルをウェブ等の別の手段で指定した後、画像を表示する部分の処理フローである。

【0 0 3 7】

まず、ユーザが表示のための操作を行う（ステップ S 7 0 0）。例えば、アプリケーション起動時の指定されたファイルの 1 回目の表示では、開いたウィンドウサイズに収まる画像サイズを計算して、当該ウィンドウサイズに一致する画像サイズで表示するようにする。そして、その後に本ステップ S 7 0 0 でユーザの操作を受け付けるようにしてもよい。

【0 0 3 8】

次に、上記ステップ S 7 0 0 でユーザ操作により新たに必要になった packet を全て算出する（ステップ S 7 0 1）。その後、ステップ S 7 0 1 で算出した新規に必要な packet のデータについてサーバに対して要求を発行する（ステップ S 7 0 2）。次いで、要求した全 packet のデータをサーバから受信し、クライアント・コンピュータ上にキャッシュする（ステップ S 7 0 3）。

【 0 0 3 9 】

さらに、高速読み出しが可能なようにキャッシュ形式で管理している符号データを複数の J P E G 2 0 0 0 符号データ形式に変換することによって符号データの作成を行う（ステップ S 7 0 4）。この符号データの作成処理の詳細については後述する。その後、ステップ S 7 0 4 で作成した J P E G 2 0 0 0 符号データをデコードして表示する（ステップ S 7 0 5）。そして、ユーザが終了を要求したか否かを判定する（ステップ S 7 0 6）。その結果、ユーザが終了を要求した場合（Y e s）、本アプリケーションを終了する。一方、ユーザが終了を要求せずに他の表示要求を行った場合（N o）、ステップ S 7 0 1 に戻って上記処理を実行する。

【 0 0 4 0 】

以下では、上記クライアント・コンピュータ上のアプリケーションでの処理手順の具体的な処理内容について説明する。

【 0 0 4 1 】

まず、クライアント・アプリケーションが起動した直後で、このアプリケーションの初期画像表示サイズを 1 2 8 × 1 2 8 画素とし、このサイズに入る画像データを表示することを考える。この場合、ステップ S 7 0 2 で算出される必要な packet は、最低解像度の画像サイズであって S N R が最大のデータを表示することを考えると、図 6 B 内のタイル 0 からタイル 1 5 までの全 1 6 タイルにおける、「Lay0/Res0/Com0/Pos0」、「Lay1/Res0/Com0/Pos0」、「Lay2/Res0/Com0/Pos0」で示す 3 × 1 6 個の packet が必要な packet である。

【 0 0 4 2 】

そこで、クライアント・プログラムは、これらの packet をサーバ・プログラムに要求する。一方、サーバ・プログラムは、この全タイルの 3 × 1 6 個の packet を切り出す処理を行う。その後、クライアント・プログラムは、これらの packet を受信した後デコードして表示を行う。

【 0 0 4 3 】

次に、図 7 のステップ S 7 0 4 における符号データ作成についての詳細な説明を図 8 ～ 1 0 を用いて行う。図 8 A は、図 7 におけるステップ S 7 0 4 符号デー

タ作成処理の概要を説明するためのフローチャートである。まず、サーバから受信した必要なpacketに関するJ P E G 2 0 0 0符号データのメインヘッダを解析する(ステップS 8 0 0)。次に、ステップS 8 0 0におけるメインヘッダを解析した結果から、符号データ内のタイルの個数を計算する(ステップS 8 0 1)。本実施形態においては、タイルの個数の計算は、以下に示す式(1)で行うものとする。

【0044】

$$\begin{aligned} Xtiles &= (Xsiz + XTsiz - 1) \div XTsiz \\ Ytiles &= (Ysiz + YTtiz - 1) \div YTtiz \\ Tiles &= Xtiles \times Ytiles \end{aligned} \quad (1)$$

そして、式(1)で求められたタイルの個数だけ、ステップS 8 0 2以下の処理が行われる。

【0045】

ステップS 8 0 2は、各タイルの符号変換処理を行う部分である。この処理の詳細については後述する。そして、全タイルについての符号変換処理が終了したか否かを判断する(ステップS 8 0 3)。その結果、未処理のタイルがある場合(N o)、ステップS 8 0 2に戻る。一方、全タイルについて符号変換処理が終了した場合(Y e s)、本処理を終了してステップS 7 0 5に進む。

【0046】

図8Bは、ステップS 8 0 2における符号変換処理を詳細に説明するためのフローチャートである。まず、今回処理するタイルがフル充填、すなわち全packetデータがキャッシュされているか否かを判断する(ステップS 8 0 4)。その結果、フル充填されている場合(Y e s)、ステップS 8 0 5に進み、それ以外の場合(N o)、ステップS 8 0 7に進む。

【0047】

ステップS 8 0 5では、フル充填状態の符号データを作成したか否かを判断する。その結果、まだ作成されていない場合(N o)はステップS 8 0 6に進み、作成済みの場合(Y e s)は当該符号変換処理を終了する。ステップS 8 0 6では、完全符号データを作成したことを示すフラグをセットする。そして、実際の

キャッシュデータから符号データを作成する「符号ファイル作成」処理が行われる（ステップS807）。この符号ファイル作成処理の詳細については後述する。

【0048】

上述したように、1度フル充填されたキャッシュデータから符号ファイルを生じた後は、再度符号ファイル作成処理（ステップS807）を行わないように制御することにより、キャッシュが充填されて行くに連れて、符号ファイル作成処理を高速に行うことができる。

【0049】

図9は、図8に示すステップS807の符号ファイル作成処理を詳細に説明するためのフローチャートである。まず、タイルに含まれるpacketの数をiPMaxにセットする（ステップS900）。次に、「Tile part Header」を仮出力する（ステップS901）。ここで仮出力するのは、この「Tile part Header」にはタイルの符号長を格納するフィールドがあり、このタイルの符号長は以下の処理を終了した時点で計算できる値であるためである。

【0050】

次に、変数iP、iLに初期値として「0」をセットする（ステップS902、S903）。ここで、変数iPはpacketのIDを示す変数であり、変数iLはタイルの符号長を示す変数である。次に、変数iPで示されるpacketがキャッシュされているか否かを判断する（ステップS904）。その結果、変数iPがキャッシュされている場合（Yes）はステップS905へ進み、変数iPがキャッシュされていない場合（No）はステップS907に進む。

【0051】

ステップS907では、該当するpacketデータが無いことを示す「Zero Length Packet」のコード（以下、「ZLPコード」と略す。）を出力する。そして、このZLPコードのバイト数である「1」を変数iLに加える（ステップS908）。一方、ステップS905では該当するpacketのデータを出力する。そして、そのバイト数を変数iLに足し込んで更新する（ステップS906）。

【0052】

ステップ S 9 0 6 及び S 9 0 8 の処理の後、変数 iP をインクリメントする（ステップ S 9 0 9）。そして、全 packet について処理したかを判断する（ステップ S 9 1 0）。その結果、まだ全 packet を処理していない場合（N o）、ステップ S 9 0 4 に戻って上記処理を行う。一方、全 packet を処理した場合（Y e s）、変数 iL から「Tile part Header」である「S O T マーカ」を更新して出力し（ステップ S 9 1 1）、当該符号ファイル生成処理を終了する。そして、前述したように当該符号ファイルを汎用の J P E G 2 0 0 0 デコーダでデコードし、デコードされた画像データを表示する（ステップ S 7 0 5）。

【0053】

すなわち、上述したように、本実施形態に係るクライアントは、サーバが管理する符号データのうち、断片的な第 1 の符号データを格納した格納手段（例えば、ハードディスク 2 0 6）を備え、J P E G 2 0 0 0 符号データを作成する符号データ作成装置として機能する。本実施形態に係るクライアントでは、まず、当該クライアントにおいて J P E G 2 0 0 0 符号データの作成に必要な符号データと、予めハードディスク 2 0 6 等の格納手段に格納された第 1 の符号データ（例えば、現在表示部 2 0 3 に表示されている画像データについての符号データ）とから、不足する第 2 の符号データ（例えば、表示部 2 0 3 で拡大表示するために必要な画像データについての符号データであって、第 1 の符号データには含まれていないもの）を算出する。そして、サーバに対して、算出された第 2 の符号データを要求し、サーバから、第 2 の符号データを取得し、取得された第 2 の符号データをハードディスク 2 0 6 等の格納手段に格納する。次いで、取得された第 2 の符号データのヘッダ情報を解析して、対象となる符号データを複数の独立した符号データに分割する。さらに、分割された単位毎に、独立した符号データを構成する全ての符号データがハードディスク 2 0 6 等の格納手段に格納されているか否かを判定する。そして、独立した符号データを構成する全ての符号データが格納されていない場合、格納されていない部分にダミー符号データを格納し、格納された符号データを J P E G 2 0 0 0 符号データとして出力することを特徴とする。

【0054】

また、上記クライアント（符号データ作成装置）では、符号データが、パケット単位で取り扱われることを特徴とする。さらに、上記クライアント（符号データ作成装置）では、ダミー符号データが、J P E G 2 0 0 0 で規定されている zero length packet データであることを特徴とする。さらにまた、上記サーバとクライアントが、ネットワークを介して互いに通信可能であることを特徴とする。

【 0 0 5 5 】

さらに、上記クライアント（符号データ作成装置）は、画像データを表示する表示手段（例えば、表示部 2 0 3）をさらに備え、上記第 1 の符号データが当該画像データの符号データであって、表示部 2 0 3 等の表示手段に表示された画像データの表示領域の移動又は拡大表示に応じて、上記第 2 の符号データの算出において必要となる符号データを設定する。また、上記クライアント（符号データ作成装置）は、出力された上記 J P E G 2 0 0 0 符号データをしてデコードし、デコードされた画像データを表示部 2 0 3 等の表示手段に画面表示することを特徴とする。

【 0 0 5 6 】

ここで、具体例を用いて、上記フローチャートで示した符号ファイル作成処理を説明する。図 1 0 は、本発明の第 1 の実施形態における初期画面での符号ファイル処理後の各タイルの符号ファイルの構成を示す図である。ここで、クライアント・コンピュータから要求される packet は、前述のとおり全タイルの「Lay0/Res0/Com0/Pos0」、「Lay1/Res0/Com0/Pos0」、「Lay2/Res0/Com0/Pos0」である。

【 0 0 5 7 】

従って、図 1 0 に示すように、全 1 6 個のタイルに対して、符号 1 0 0 1 で示すように上記 3 つの packet データ以外の packet については Z L P が格納された符号データが作成される。また、各タイル毎に作成される符号データのメインヘッダ部分は、メインヘッダ内の S I Z マーカコード内の Xsiz、Ysiz の各フィールドがタイルのサイズである 2 5 6 に変更されている。これにより、各タイル毎に独立した符号ファイルを作成することができる。

【 0 0 5 8 】

すなわち、本実施形態に係るクライアント（符号データ作成装置）では、符号データ（画像データ）を所定サイズのタイル単位（例えば、式（1）に記載の方法で算出された単位）で分割することを特徴とする。また、同クライアントでは、独立した符号データのそれぞれのヘッダ情報を、分割されたタイル単位のサイズに変更することを特徴とする。

【 0 0 5 9 】

次に、ステップ S 7 0 1 でユーザから拡大表示が指示された場合について説明する。この拡大表示では、まず初期画面より 2 倍に拡大するものとする。この場合、表示ウィンドウサイズは 1 2 8 × 1 2 8 画素であるとする、表示できるタイルは 4 個になる。例えば、図 6 A に示す Tile5、Tile6、Tile9、Tile10 の 4 個のタイルの Resolution1（画像サイズ：6 4 × 6 4 画素）を表示するものとする。この場合、ステップ S 7 0 2 では、Tile5、Tile6、Tile9、Tile10 の 4 つのタイルの「Lay0/Res1/Com0/Pos0」、「Lay1/Res1/Com0/Pos0」、「Lay2/Res1/Com0/Pos0」がクライアント・コンピュータからサーバに対して要求される。

【 0 0 6 0 】

さらに、ユーザから 2 倍に拡大する要求が発行された場合、表示ウィンドウサイズを 1 2 8 × 1 2 8 画素から 2 5 6 × 2 5 6 画素に変更すると、上記 4 タイルに対してさらに、「Lay0/Res2/Com0/Pos0」、「Lay1/Res2/Com0/Pos0」、「Lay2/Res2/Com0/Pos0」が要求される。

【 0 0 6 1 】

その後、さらに、2 倍に拡大する要求がユーザからあったとする。この場合、表示領域のウィンドウサイズを変更しないとすると、タイルが 1 つ分しか表示することができない。そこで、Tile5 のみを表示するとすると、Tile5 の「Lay0/Res3/Com0/Pos0」、「Lay1/Res3/Com0/Pos0」、「Lay2/Res3/Com0/Pos0」をサーバに要求し表示することになる。これにより、Tile5 は完全にキャッシュが充填されたことになる。図 1 1 は、第 1 の実施形態における符号データの各タイル毎の符号ファイルの状態の一例を説明するための図である。

【 0 0 6 2 】

図 1 1 に示すように、Tile0、Tile1、Tile2、Tile3、Tile4、Tile7、Tile8、T

ile11、Tile12、Tile13、Tile14及びTile15の各タイルの符号ファイルは、図10に示したものと同様に符号1001で示す状態になっている。また、Tile6、Tile9及びTile10の各タイルは符号1101に示す状態に、Tile5は符号1102に示す状態になっている。この状態になるまでは、ステップS805における完全符号データを作成済みか否かの判断により全てステップS806に進むようになるが、Tile5の符号変換処理後はステップS805の判断後、ステップS806には進まずに処理を終了するようになる。これにより、Tile5については、これ以降、符号変換処理を行う必要がなくなる。これにより、キャッシュが充填されるほど符号ファイル作成処理を高速にすることができる。すなわち、本実施形態によれば、デコード／表示する領域をタイル単位で行う場合、デコード／表示に必要なタイルのみに対して処理を行うことができる。

【0063】

また、従来は、キャッシュファイルから1つのJPEG2000符号データファイルを作成する場合、表示するタイル毎のマルチスレッド化は困難であった。これは、1つの符号データファイルを作成するため、ここで処理をシリアル化することが必須であったためである。しかし、本実施形態のように各タイル毎の符号データファイルを作成することにより、表示するタイル毎に必要なpacket要求からキャッシュファイルの生成、キャッシュファイルから符号データの作成、符号データのデコード及び表示までを各タイル毎スレッドとしてマルチスレッド化することができる。従って、これまでのシングルスレッドの場合に比べて、マルチスレッド化することによる高速化も図ることができる。

【0064】

さらに、ファイルI/Oの処理時間がかかるようなシステムにおいては、1つの大きな符号データファイルを作成する代わりに、小さな複数の符号ファイルを作成することで、ファイルI/Oによる時間のロスを最小限にすることも可能となる。

【0065】

<第2の実施形態>

上述した第1の実施形態では、キャッシュされている全てのタイルに対して符

号変換処理を行うことを説明したが、本実施形態では、表示に必要な部分のタイルのみ符号変換処理を行うことにより、第1の実施形態よりもさらに高速に符号変換処理を行う場合について説明する。

【0066】

図12は、本発明の第2の実施形態に係る符号データ作成処理の概要を説明するためのフローチャートである。尚、上述した第1の実施形態における図8Aのフローチャートと同じ処理の部分には同一の番号を付けている。ここでは、第1の実施形態と異なる部分のみを説明する。

【0067】

ステップS801のタイル個数の計算の後、ステップS701でユーザが操作したコマンドを解析する（ステップS1200）。そして、この解析結果を使用して、これから処理しようとしているタイルが今回の表示で表示されるタイルか否かを判断する（ステップS1201）。その結果、ここで表示に使用しないタイルであると判断された場合（No）、ステップS803に進む。一方、表示に使用するタイルであると判断された場合（Yes）、ステップS1202に進む。

【0068】

ステップS1202では、ステップS703で新たにpacketを要求した場合に、今回処理しようとしているタイルのpacketを取得したか否かを判断する。その結果、新規packetを取得している場合（Yes）はステップS802に進み、新規packetを取得していない場合（No）はステップS803に進む。尚、これ以降の処理手順については、第1の実施形態で説明した手順と同様である。これらの処理により、表示に必要なタイルであって、まだ完全に充填されていないタイルのみ符号変換処理を行うことができるため、当該符号化処理を第1の実施形態よりも高速に行うことができる。

【0069】

<その他の実施形態>

尚、本発明は、複数の機器（例えば、ホストコンピュータ、インタフェース機器、リーダ、プリンタ等）から構成されるシステムに適用しても、一つの機器か

らなる装置（例えば、複写機、ファクシミリ装置等）に適用してもよい。

【0 0 7 0】

また、本発明の目的は、前述した実施形態の機能を実現するソフトウェアのプログラムコードを記録した記録媒体（または記憶媒体）を、システムあるいは装置に供給し、そのシステムあるいは装置のコンピュータ（またはCPUやMPU）が記録媒体に格納されたプログラムコードを読み出し実行することによっても、達成されることは言うまでもない。この場合、記録媒体から読み出されたプログラムコード自体が前述した実施形態の機能を実現することになり、そのプログラムコードを記録した記録媒体は本発明を構成することになる。また、コンピュータが読み出したプログラムコードを実行することにより、前述した実施形態の機能が実現されるだけでなく、そのプログラムコードの指示に基づき、コンピュータ上で稼働しているオペレーティングシステム（OS）などが実際の処理の一部または全部を行い、その処理によって前述した実施形態の機能が実現される場合も含まれることは言うまでもない。

【0 0 7 1】

さらに、記録媒体から読み出されたプログラムコードが、コンピュータに挿入された機能拡張カードやコンピュータに接続された機能拡張ユニットに備わるメモリに書き込まれた後、そのプログラムコードの指示に基づき、その機能拡張カードや機能拡張ユニットに備わるCPUなどが実際の処理の一部または全部を行い、その処理によって前述した実施形態の機能が実現される場合も含まれることは言うまでもない。

【0 0 7 2】

本発明を上記記録媒体に適用する場合、その記録媒体には、先に説明したフローチャートに対応するプログラムコードが格納されることになる。

【0 0 7 3】

【発明の効果】

以上説明したように、本発明によれば、クライアントにキャッシュされている断片化された符号データと、サーバから必要な分だけ受信した断片化された符号データとを用いて、クライアントにおいて汎用JPEG 2000デコーダで使用

可能な符号データであって、当該符号データのデコード及び画像データの表示処理を高速に行うことができる符号データを好適に作成することができる。

【図面の簡単な説明】

【図 1】

本発明の第 1 の実施形態に係るサーバ及びクライアントを含むネットワークシステムの構成を示す概要図である。

【図 2】

図 1 のサーバ・コンピュータ 1 0 1、1 0 3 又はクライアント・コンピュータ 1 0 2 a、1 0 2 b の各コンピュータシステムのハードウェア構成の一例を示す図である。

【図 3】

Layer-resolution level-component-position progression (S N R Progression) に沿って記録された J P E G 2 0 0 0 ファイルの構成を示す概念図である。

【図 4】

J P E G 2 0 0 0 の解像度スケーラビリティを説明する図である。

【図 5】

第 1 の実施形態におけるサーバとクライアント間での packet 単位のデータのリクエスト及びレスポンスを説明するための概念図である。

【図 6 A】

サーバ・コンピュータに格納されている原画像データをタイルに分割した例を示す図である。

【図 6 B】

図 6 A で示したタイル分割された画像データに関する J P E G 2 0 0 0 符号データのデータ構造を示す図である。

【図 7】

第 1 の実施形態におけるクライアント・コンピュータ上のアプリケーションでの処理手順の概要を説明するためのフローチャートである。

【図 8 A】

図 7 におけるステップ S 7 0 4 符号データ作成処理の概要を説明するためのフローチャートである。

【図 8 B】

ステップ S 8 0 2 における符号変換処理を詳細に説明するためのフローチャートである。

【図 9】

図 8 に示すステップ S 8 0 7 の符号ファイル作成処理を詳細に説明するためのフローチャートである。

【図 1 0】

本発明の第 1 の実施形態における初期画面での符号ファイル処理後の各タイルの符号ファイルの構成を示す図である。

【図 1 1】

第 1 の実施形態における符号データの各タイル毎の符号ファイルの状態の一例を説明するための図である。

【図 1 2】

本発明の第 2 の実施形態に係る符号データ作成処理の概要を説明するためのフローチャートである。

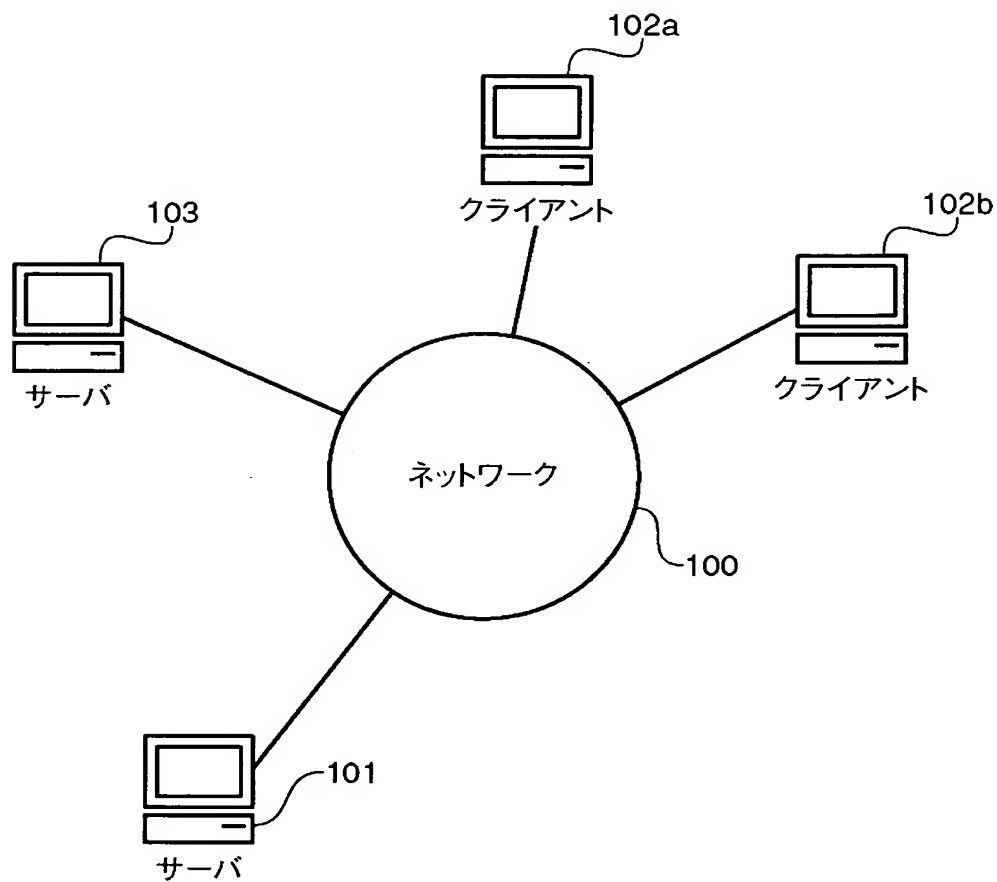
【符号の説明】

5 0 1 サーバ

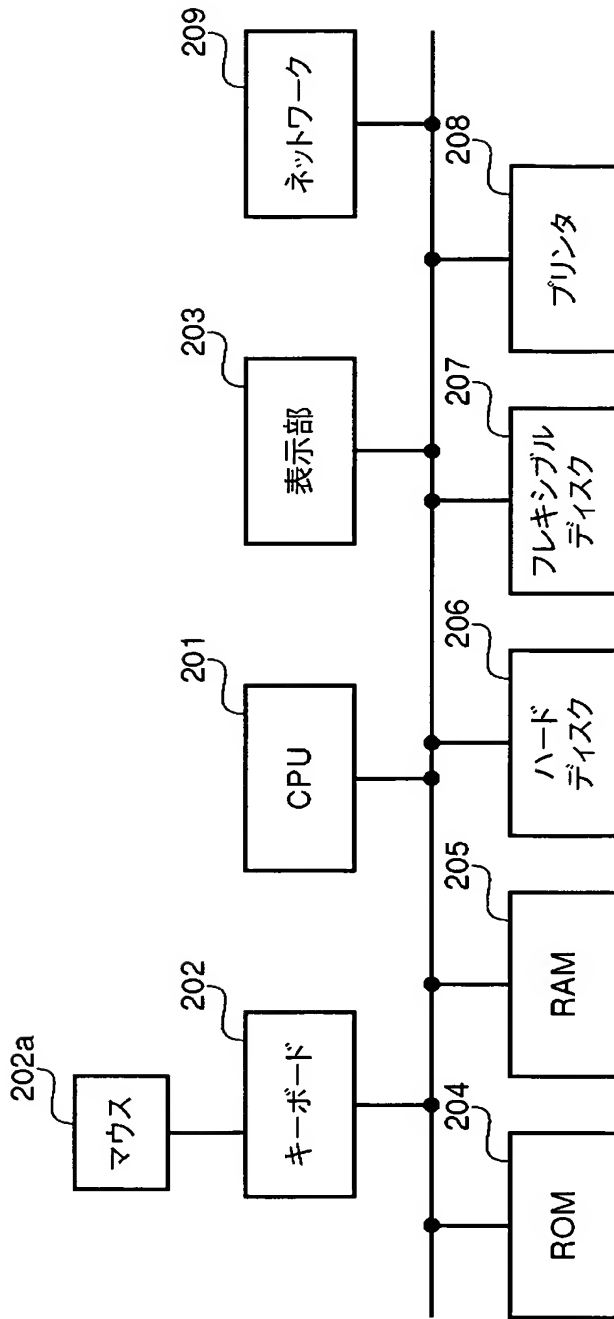
5 0 2 クライアント

【書類名】 図面

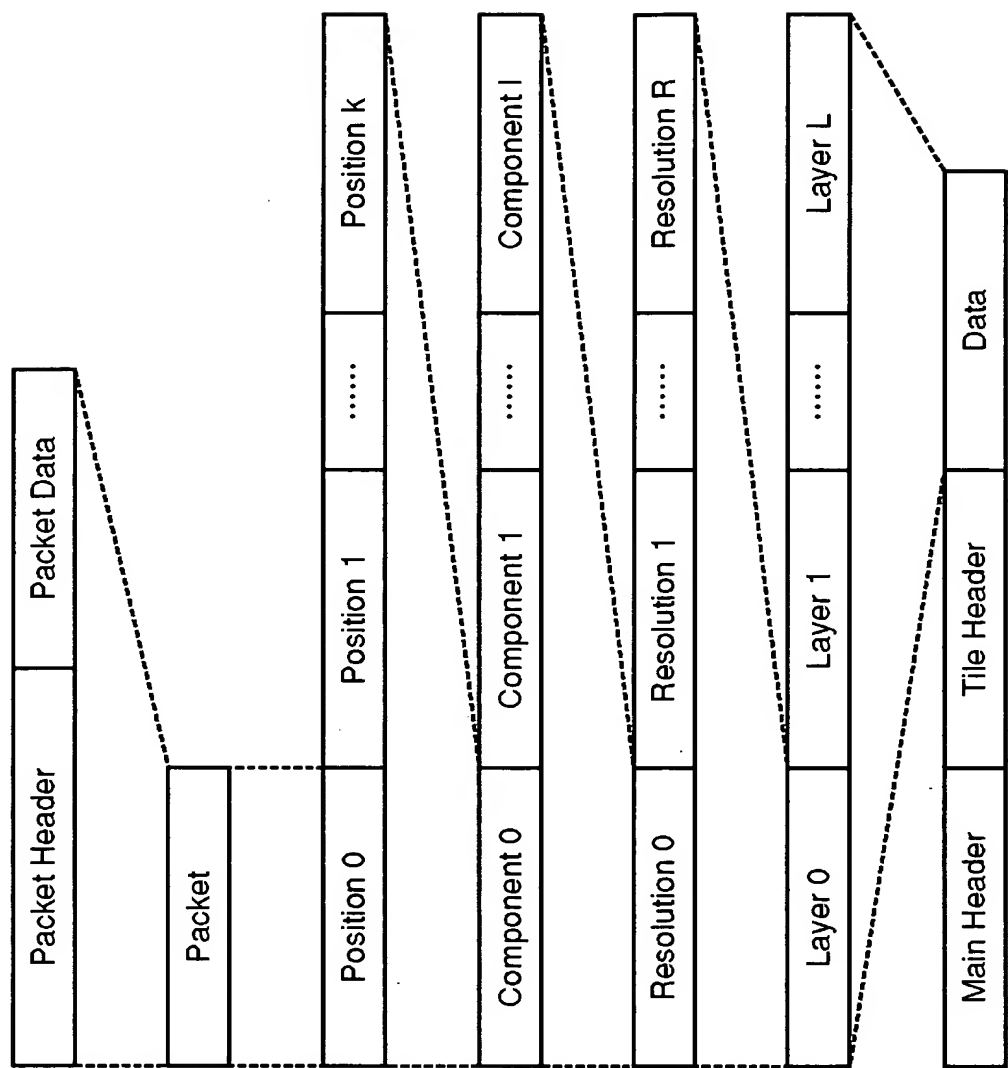
【図 1】



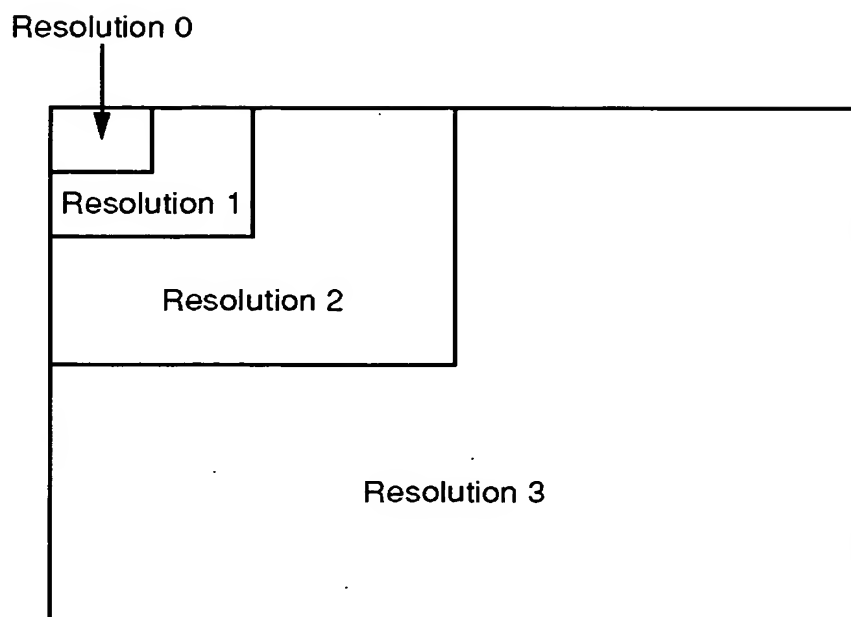
【図 2】



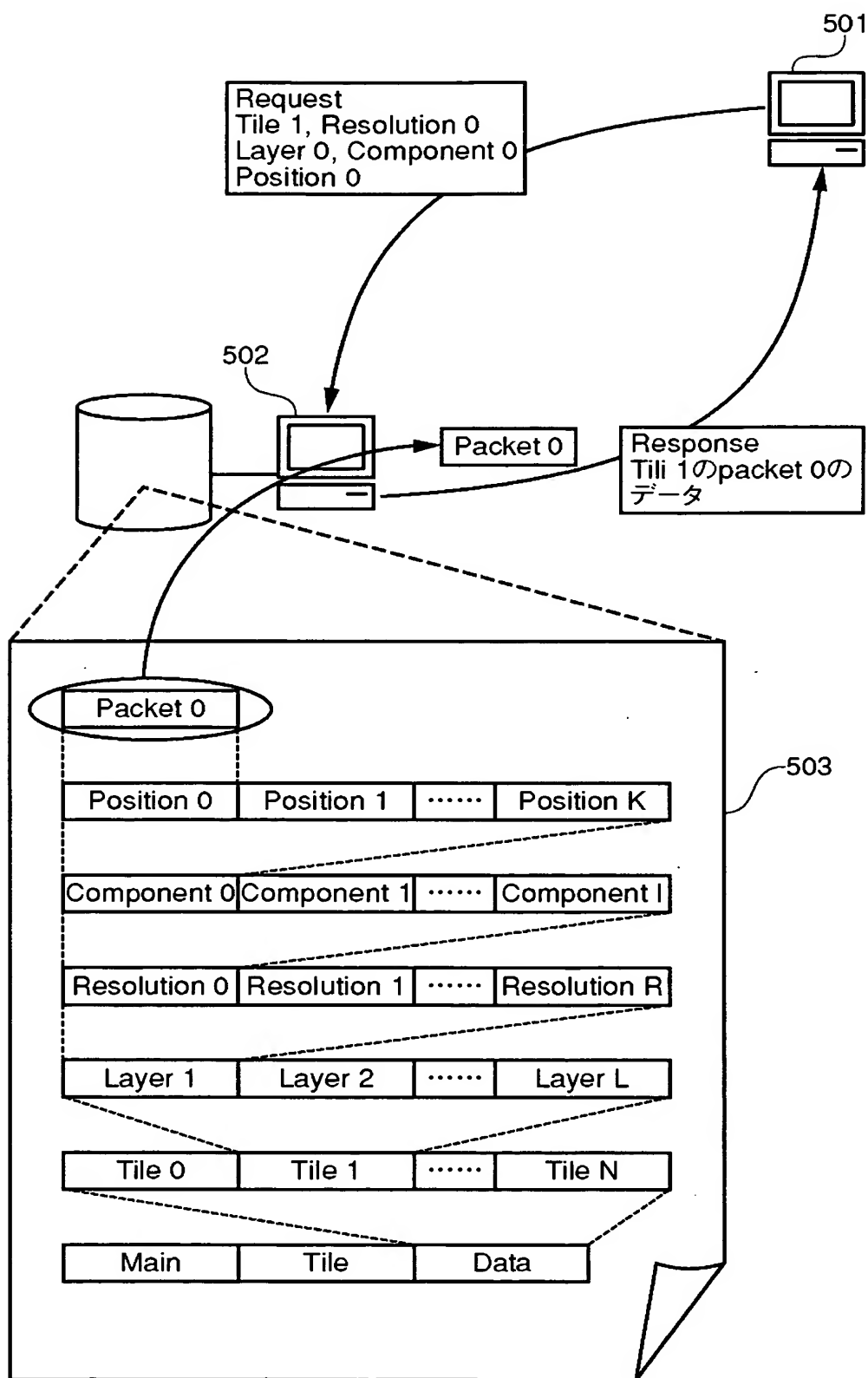
【図 3】



【図 4】



【図 5】



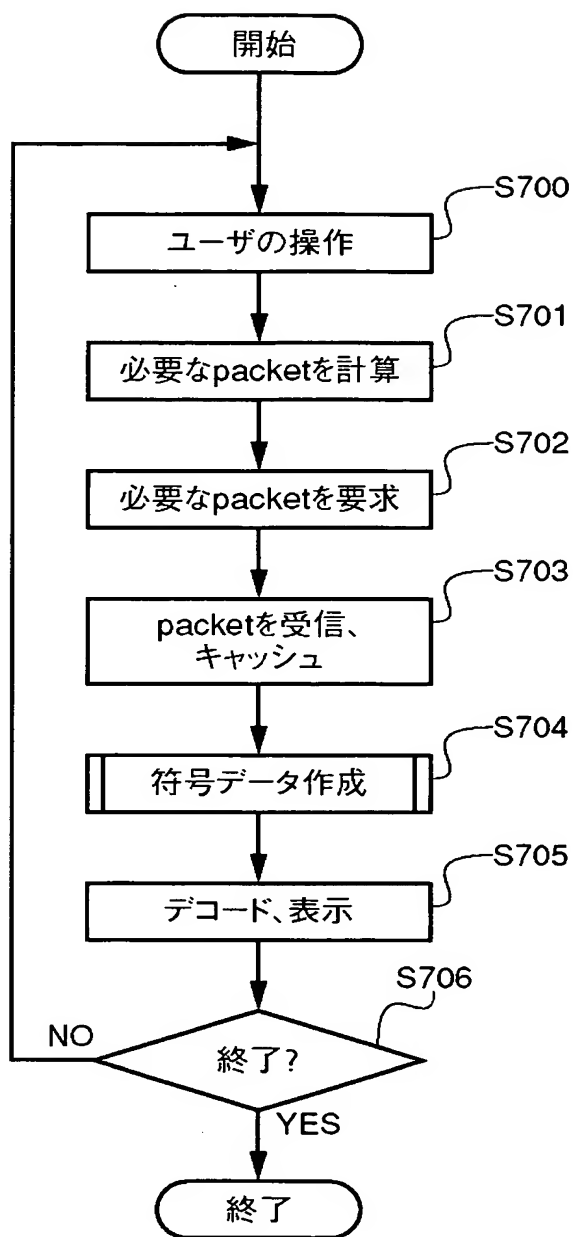
【図 6 A】

Tile 0	Tile 1	Tile 2	Tile 3
Tile 4	Tile 5	Tile 6	Tile 7
Tile 8	Tile 9	Tile 10	Tile 11
Tile 12	Tile 13	Tile 14	Tile 15

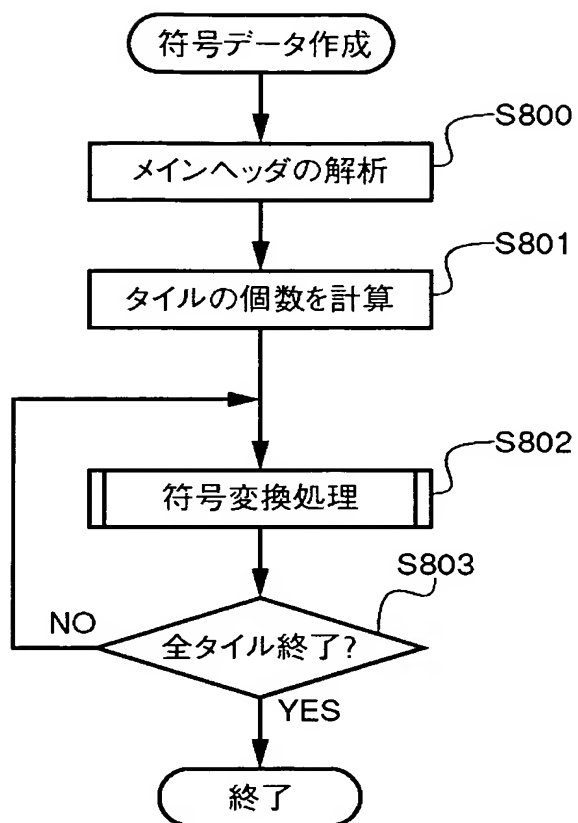
【図 6 B】

Main Header	SIZマーカ・コード	Xsize = 1024
	CODマーカ・コード	Ysize = 1024
	QCDマーカ・コード	XTsize = 256
Tile part Header 0	SOTマーカ・コード	YTsize = 256
	SODマーカ・コード	
符号化データ	Lay0/Res0/Com0/Pos0	
	Lay0/Res1/Com0/Pos0	
	Lay0/Res2/Com0/Pos0	
	Lay0/Res3/Com0/Pos0	
	Lay1/Res0/Com0/Pos0	
	Lay1/Res1/Com0/Pos0	
	Lay1/Res2/Com0/Pos0	
	Lay1/Res3/Com0/Pos0	
	Lay2/Res0/Com0/Pos0	
	Lay2/Res1/Com0/Pos0	
	Lay2/Res2/Com0/Pos0	
	Lay2/Res3/Com0/Pos0	
Tile part Header 1	SOTマーカ・コード	
	SODマーカ・コード	
符号化データ		
:		
Tile part Header 15	SOTマーカ・コード	
	SODマーカ・コード	
符号化データ		

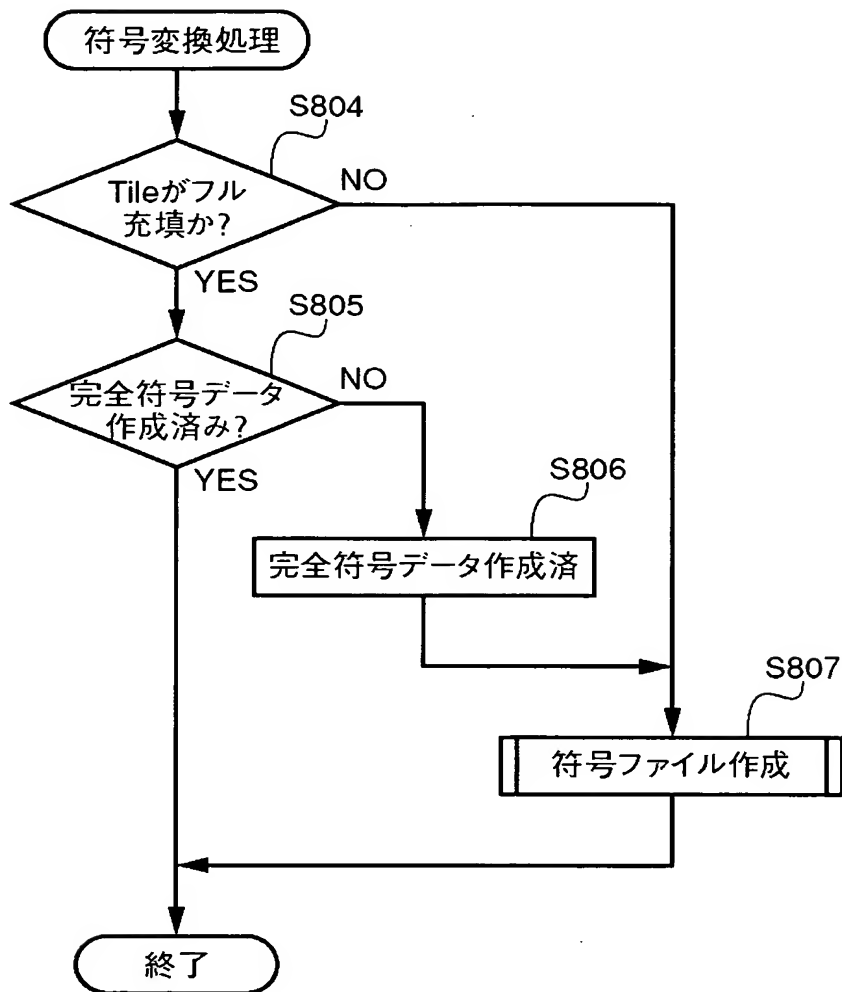
【図 7】



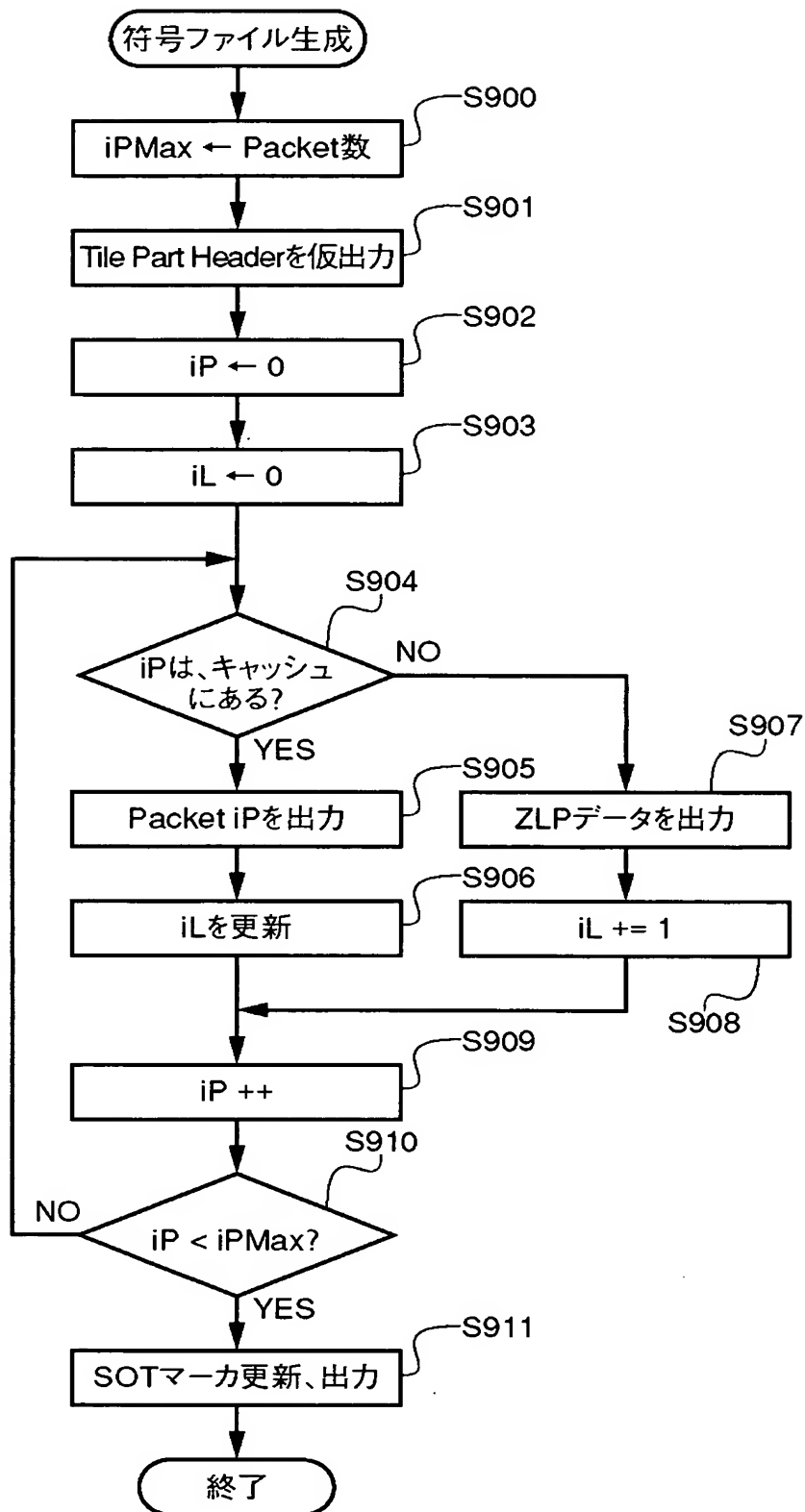
【図 8 A】



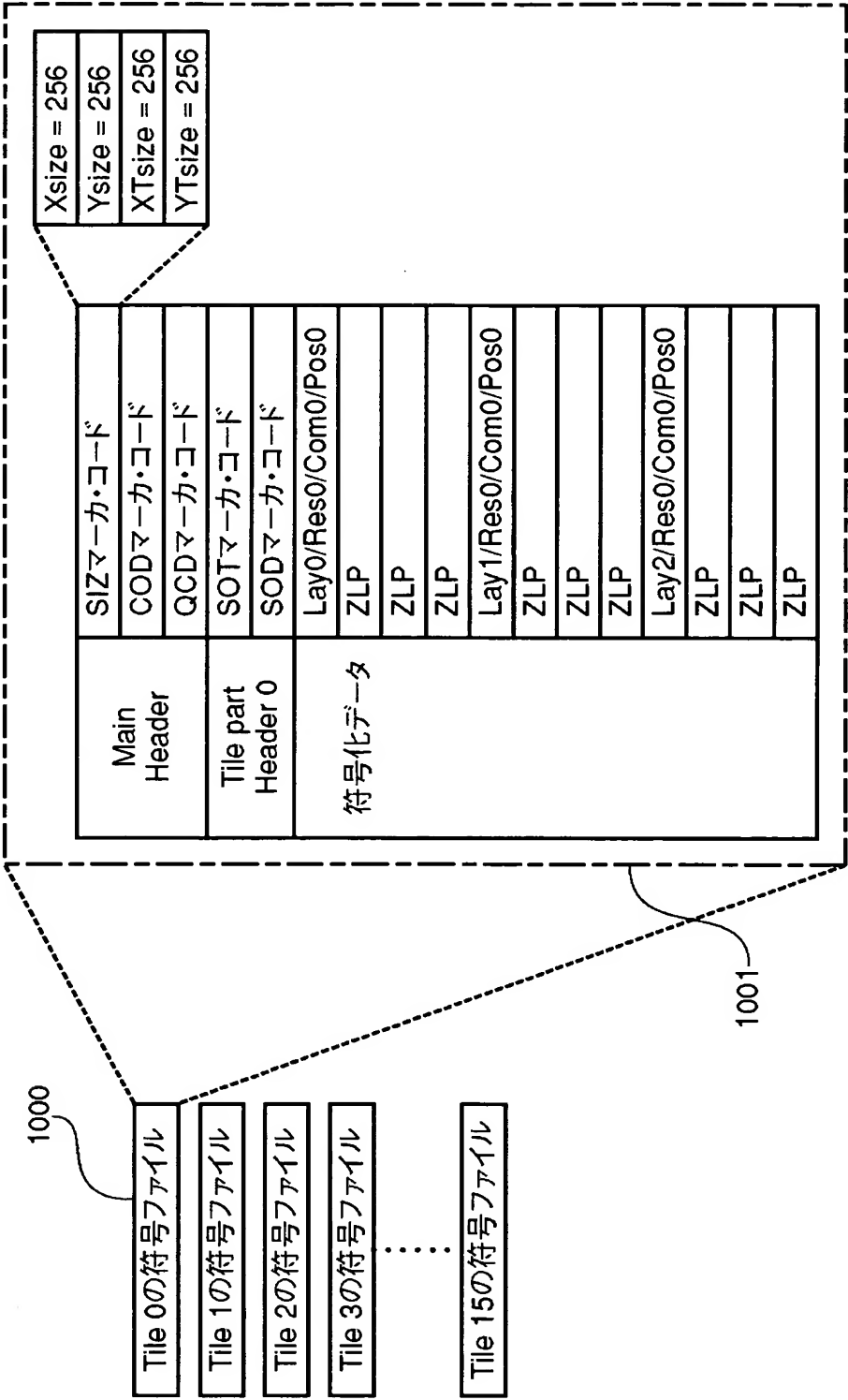
【図 8 B】



【図 9】



【図 1 0】



【図 1 1】

1102

Main Header	SIZマーカ・コード
	CODマーカ・コード
	QCDマーカ・コード
Tile part Header 0	SOTマーカ・コード
	SODマーカ・コード
符号化データ	Lay0/Res0/Com0/Pos0
	Lay0/Res1/Com0/Pos0
	Lay0/Res2/Com0/Pos0
	Lay0/Res3/Com0/Pos0
	Lay1/Res0/Com0/Pos0
	Lay1/Res1/Com0/Pos0
	Lay1/Res2/Com0/Pos0
	Lay1/Res3/Com0/Pos0
	Lay2/Res0/Com0/Pos0
	Lay2/Res1/Com0/Pos0
	Lay2/Res2/Com0/Pos0
	Lay2/Res3/Com0/Pos0

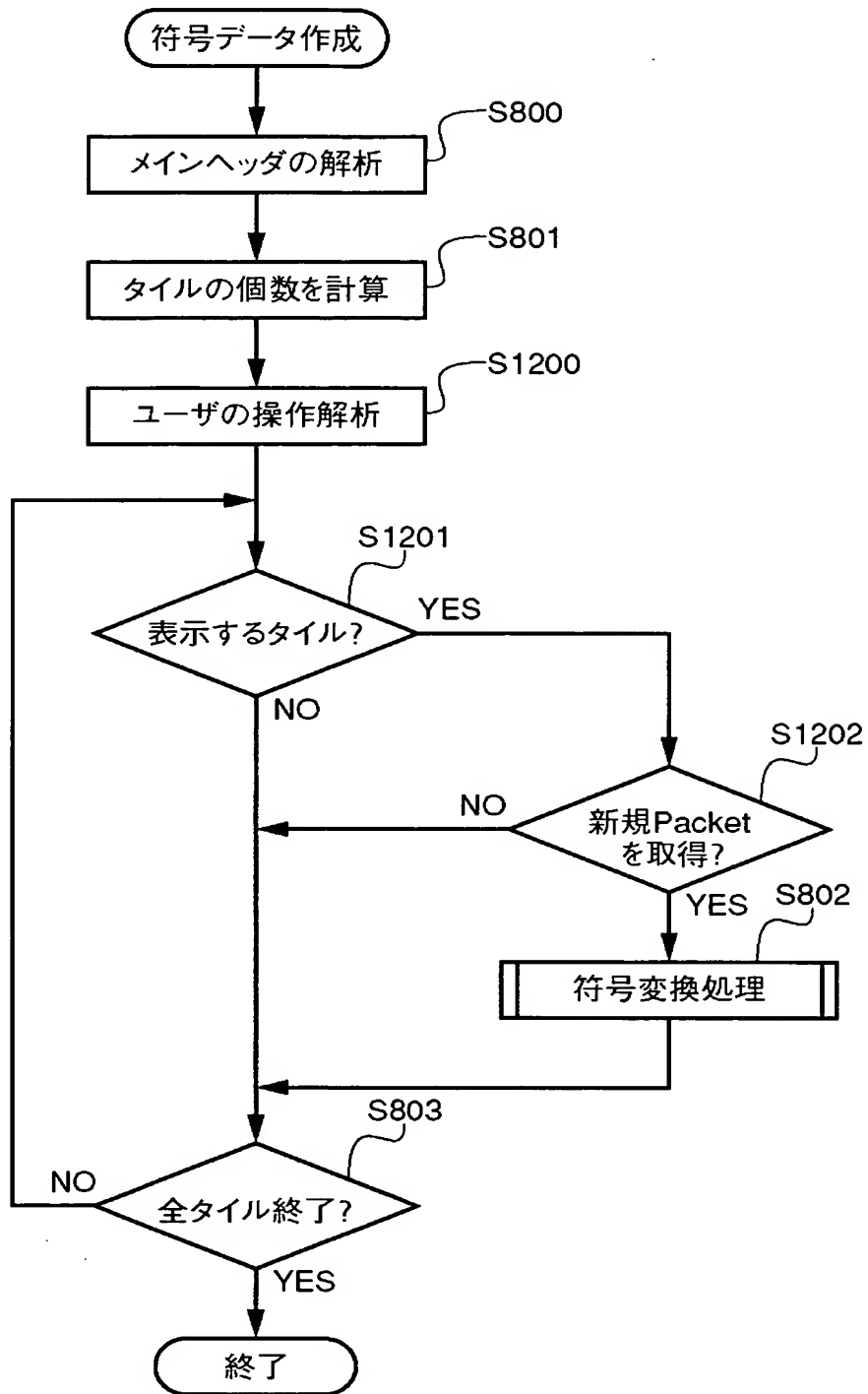
1101

Main Header	SIZマーカ・コード
	CODマーカ・コード
	QCDマーカ・コード
Tile part Header 0	SOTマーカ・コード
	SODマーカ・コード
符号化データ	Lay0/Res0/Com0/Pos0
	Lay0/Res1/Com0/Pos0
	Lay0/Res2/Com0/Pos0
	ZLP
	Lay1/Res0/Com0/Pos0
	Lay1/Res1/Com0/Pos0
	Lay1/Res2/Com0/Pos0
	ZLP
	Lay2/Res0/Com0/Pos0
	Lay2/Res1/Com0/Pos0
	Lay2/Res2/Com0/Pos0
	ZLP

1001

Main Header	SIZマーカ・コード
	CODマーカ・コード
	QCDマーカ・コード
Tile part Header 0	SOTマーカ・コード
	SODマーカ・コード
符号化データ	Lay0/Res0/Com0/Pos0
	ZLP
	ZLP
	ZLP
	Lay1/Res0/Com0/Pos0
	ZLP
	ZLP
	ZLP
	Lay2/Res0/Com0/Pos0
	ZLP
	ZLP
	ZLP

【図 12】



【書類名】 要約書

【要約】

【課題】 クライアントにキャッシュされている断片化された符号データと、サーバから必要な分だけ受信した断片化された符号データとを用いて、クライアントにおいて汎用 J P E G 2 0 0 0 デコーダで使用可能な符号データを好適に作成することができる符号データ作成方法を提供する。

【解決手段】 クライアントは、サーバが管理する符号データのうち、第 1 の符号データを格納しており、J P E G 2 0 0 0 符号データの作成に必要となる符号データと第 1 の符号データとから、不足する第 2 の符号データを算出する。そして、サーバから第 2 の符号データを取得し、そのヘッダ情報を解析して、符号データを複数の独立した符号データに分割する。さらに、分割された単位毎に、独立した符号データの全てのデータが格納されていない場合、ダミー符号データを格納し、その符号データを J P E G 2 0 0 0 符号データとする。

【選択図】 図 5

特願 2 0 0 2 - 3 5 6 7 3 8

出 願 人 履 歴 情 報

識別番号

[0 0 0 0 0 1 0 0 7]

1 . 変更年月日

1 9 9 0 年 8 月 3 0 日

[変更理由]

新規登録

住 所

東京都大田区下丸子 3 丁目 3 0 番 2 号

氏 名

キャノン株式会社